Web-based Supplementary Materials for Population Intervention Causal Effects Based on Stochastic Interventions by Iván Díaz Muñoz and Mark van der Laan.

A Review of Efficiency Estimation in Semiparametric Models

The objective of this section is to provide an intuitive explanation of certain elements of efficient estimation in semiparametric models. We do not pretend to give a comprehensive or rigorous review, instead we intend to provide the non trained reader with the basic intuition for understanding why the methods described in the paper work. Careful and detailed definitions of the concepts described here, and rigorous proofs of most of the claims can be found in **?** and **?**.

A.1 Asymptotically Linear Estimators

Let $X \sim P_0 \in \mathcal{M}$, where $\mathcal{M}$ is a statistical (semi or non parametric) model, and let $\Psi : \mathcal{M} \mapsto \mathbb{R}$ be a parameter defined as a mapping that takes elements in the model and maps them into the reals (e.g., the mean $\Psi(P) = \int x dP(x)$). An estimator ψ_n of $\psi_0 = \Psi(P_0)$ is called asymptotically linear if there exist a function $IC : \mathcal{X} \times \mathcal{M} \mapsto \mathbb{R}$ such that $IC(\cdot, P_0) \in L_2(P_0)$, $\int IC(x, P_0) dP_0(x) = 0$ (**?**), and

$$\psi_n - \psi_0 = \frac{1}{n}\sum_{i=1}^{n} IC(X_i, P_0) + o_P(n^{-1/2}).$$

The function IC is called the influence function of the estimator, and plays an important role in estimation and inference, since it defines the asymptotic variance of the estimator. From the central limit theorem, we conclude that if ψ_n is asymptotically linear with influence curve IC, then $\sqrt{n}(\psi_n - \psi_0) \xrightarrow{d} N\{0, P_0 IC^2(\cdot, P_0)\}$.

A.2 Efficiency

Consider a family of parametric submodels $\mathcal{M}_\epsilon = \{P_\epsilon : \epsilon\} \subset \mathcal{M}$ that covers $\mathcal{M}$ and satisfies $P_{\epsilon=0} = P_0$. A typical choice of family of parametric submodels is $\{\{p_\epsilon(x) = [1 + \epsilon s(x)] p_0(x) : \epsilon\} : P_0 s = 0\}$, where each parametric submodel is indexed by a function s, which is also its score. The tangent space is defined as the closed linear span of the scores of all parametric submodels. A parameter

Ψ is called pathwise differentiable if there exists a function ν such that for each submodel

$$\left.\frac{d\Psi(P_\epsilon)}{d\epsilon}\right|_{\epsilon=0} = P_0 \nu s.$$

The function ν is called a gradient of the pathwise derivative. The only gradient D that is an element of the tangent space is called the efficient influence function, and corresponds with the influence function of any regular asymptotically linear (RAL) efficient estimator, i.e., any RAL estimator whose asymptotic variance equals the efficiency bound (**?**), which is the semiparametric generalization to the Cramer-Rao lower bound. The efficiency bound is then equal to $E_{P_0} D^2(X)$.

The efficient influence function has been used by several authors (**????**) to construct RAL efficient estimators. The basic idea to optimally estimate the parameter of interest is to find estimators that solve the efficient influence curve equation. The properties of estimators that solve a system of equations have been extensively studied in the literature and are provided by the theory of M-estimators. Important references in M-estimation include **?**, **?**, **?** and **?**.

A.3 Proofs

Proof. Result 1. First of all, notice that the nonparametric estimator of ψ_0 is given by

$$\begin{aligned}
\hat{\Psi}(P_n) &= \sum_{y\in\mathcal{Y}}\sum_{a\in\mathcal{A}}\sum_{w\in\mathcal{W}} y P_n(y|a,w)P_n(a-\delta(w)|w)P_n(w) \\
&= \sum_{y\in\mathcal{Y}}\sum_{a\in\mathcal{A}}\sum_{w\in\mathcal{W}} y \frac{P_n f_{y,a,w}}{P_n f_{a,w}} P_n f_{a-\delta(w),w}, \qquad (1)
\end{aligned}$$

where $P_n = \frac{1}{n}\sum_{i=1}^n \delta_{o_i}$ is the empirical measure, $f_{y,a,w} = I(Y = y, A = a, W = w)$, $f_{a,w} = I(A = a, W = w)$, $f_{a-\delta(w),w} = I(A = a - \delta(w), W = w)$, and $I(\cdot)$ denotes the indicator function. Here Pf denotes $\int f dP$.

Recall that the efficient influence curve in a non-parametric model corresponds with the influence curve of the non-parametric estimator. This is true because the influence curve of any regular estimator is also a gradient, and a non-parametric model has only one gradient. **?** show that if $\hat{\Psi}(P_n)$ is a substitution estimator such that $\psi_0 = \hat{\Psi}(P_0)$, and $\hat{\Psi}(P_n)$ can be written as $\hat{\Psi}^*(P_n f : f \in \mathcal{F})$ for some class of functions $\mathcal{F}$ and some mapping Ψ^*, the influence curve of $\hat{\Psi}(P_n)$ is equal to

$$IC(P_0)(O) = \sum_{f\in\mathcal{F}} \frac{d\hat{\Psi}^*(P_0)}{dP_0 f}\{f(O) - P_0 f\}.$$

Applying this result to (**??**) with $\mathcal{F} = \{f_{y,a,w}, f_{a,w}, f_{a-\delta(w),w}\}$ gives the desired result. □

Proof. Result 2. Conditioning first on (A, W) and then on W we get

$$\begin{aligned}
E_{P_0} D(O|\psi_0, \bar{Q}, g) = E_{P_0}&\left[\sum_{a\in\mathcal{A}} \frac{g_0(a|W)}{g(a|W)} g(a-\delta(W)|W)\{\bar{Q}_0(a,W) - \bar{Q}(a,W)\}\right] \\
&+ E_{P_0}\left[\sum_{a\in\mathcal{A}} g_0(a-\delta(W)|W)\bar{Q}(a,W)\right] - E_{P_0}\left[\sum_{a\in\mathcal{A}} g_0(a-\delta(W)|W)\bar{Q}_0(a,W)\right],
\end{aligned}$$

which completes the proof. □

B R function `tmle.shift()`

B.1 Arguments

Argument	Description
`Y`	Outcome vector.
`A`	Treatment vector.
`W`	Covariates matrix.
`Qn`	An initial estimator of $\bar{Q}_0$ in the form of a function that takes a vector $\mathbf{A}$ and a matrix $\mathbf{W}$ and returns the vector of conditional expectations of Y given $\mathbf{A}$ and $\mathbf{W}$.
`gn`	An initial estimator g_0 that takes as input a vector $\mathbf{A}$ and a matrix $\mathbf{W}$ and returns the density of A conditional on W at points $\mathbf{A}$.
`delta`	A function of W defining the parameter of interest.
`tol`	Tolerance value for the convergence of ϵ.
`max.iter`	Maximum of iterations allowed.
`Aval`	A vector with equally spaced values indicating a partition of the support of A over which to compute Riemann sums to approximate the integrals involved in the estimation process.

Table 1: Arguments of the R function `tmle.shift`

B.2 Code

```
tmle.shift <- function(Y, A, W, Qn, gn, delta, tol = 1e-5, iter.max = 5, Aval){
  # interval partition length
  h.int <- Aval[3]-Aval[2]
  # this function takes as input initial estimator of Q and g and returns
  # their updated value
  f.iter <- function(Qn, gn, gn0d = NULL, prev.sum = 0, first = FALSE){
      # numerical integrals and equation (7)
      Qnd <- t(sapply(1:nrow(W), function(i)Qn(Aval + delta, W[i,])))
      gnd <- t(sapply(1:nrow(W), function(i)gn(Aval, W[i,])))
      gnd <- gnd/rowSums(gnd)
      if(first) gn0d <- gnd
      EQnd <- rowSums(Qnd*gnd)*h.int
      D2   <- Qnd - EQnd
      QnAW <- Qn(A, W)
      H1   <- gn(A - delta, W)/gn(A, W)
      # equation (8)
      est.equation <- function(eps){
        sum((Y - (QnAW + eps*H1)) * H1 + (Qn(A + delta, W) - EQnd) -
        rowSums(D2*exp(eps*D2 + prev.sum)*gn0d)/rowSums(exp(eps*D2 + prev.sum)*gn0d))
      }
      eps  <- uniroot(est.equation, c(-1, 1))$root
      # updated values
```

```
      gn.new   <- function(a, w)exp(eps*Qn(a + delta, w)) * gn(a, w)
      Qn.new   <- function(a, w)Qn(a, w) + eps * gn(a - delta, w)/gn(a, w)
      prev.sum <- prev.sum + eps*D2
      return(list(Qn = Qn.new, gn = gn.new, prev.sum =
                     prev.sum, eps = eps, gn0d = gn0d))
  }
  ini.out <- f.iter(Qn, gn, first = TRUE)
  gn0d    <- ini.out$gn0d
  iter = 0
  # iterative procedure
  while(abs(ini.out$eps) > tol & iter <= iter.max){
    iter = iter + 1
    new.out <- f.iter(ini.out$Qn, ini.out$gn, gn0d, ini.out$prev.sum)
    ini.out <- new.out
  }
Qnd <- t(sapply(1:nrow(W), function(i)ini.out$Qn(Aval + delta, W[i,])))
gnd <- t(sapply(1:nrow(W), function(i)ini.out$gn(Aval, W[i,])))
gnd <- gnd/rowSums(gnd)
# plug in tmle
psi.hat <- mean(rowSums(Qnd*gnd)*h.int)
# influence curve of tmle
IC      <- (Y - ini.out$Qn(A, W))*ini.out$gn(A - delta, W)/ini.out$gn(A, W) +
           ini.out$Qn(A + delta, W) - psi.hat
var.hat <- var(IC)/length(Y)
return(c(psi.hat = psi.hat, var.hat = var.hat, IC = IC))
}
```

B.3 Example

Here is an example of how to use the previous function based on the data generating mechanism presented in the simulation

```
n  <- 100
W <- data.frame(W1 = runif(n), W2 = rbinom(n, 1, 0.7))
A  <- rpois(n, lambda = exp(3 + .3*log(W$W1) - .2*exp(W$W1)*W$W2))
Y  <- rbinom(n, 1, plogis(-1 + .05*A - .02*A*W$W2 + .2*A*tan(W$W1^2) -
      .02*W$W1*W$W2 + 0.1*A*W$W1*W$W2))
fitA.0 <- glm(A ~ I(log(W1)) + I(exp(W1)):W2, family = poisson, data = data.frame(A, W))
fitY.0 <- glm(Y ~ A + A:W2 + A:I(tan(W1^2)) + W1:W2 + A:W1:W2, family =
          binomial, data = data.frame(A, W))
gn.0  <- function(A = A, W = W)dpois(A, lambda = predict(fitA.0, newdata = W,
         type = "response"))
Qn.0 <- function(A = A, W = W)predict(fitY.0, newdata = data.frame(A, W,
        row.names = NULL), type = "response")
tmle00 <- tmle.shift(Y, A, W, Qn.0, gn.0, delta=2, tol = 1e-4, iter.max = 5,
          Aval = seq(1, 60, 1))
```